

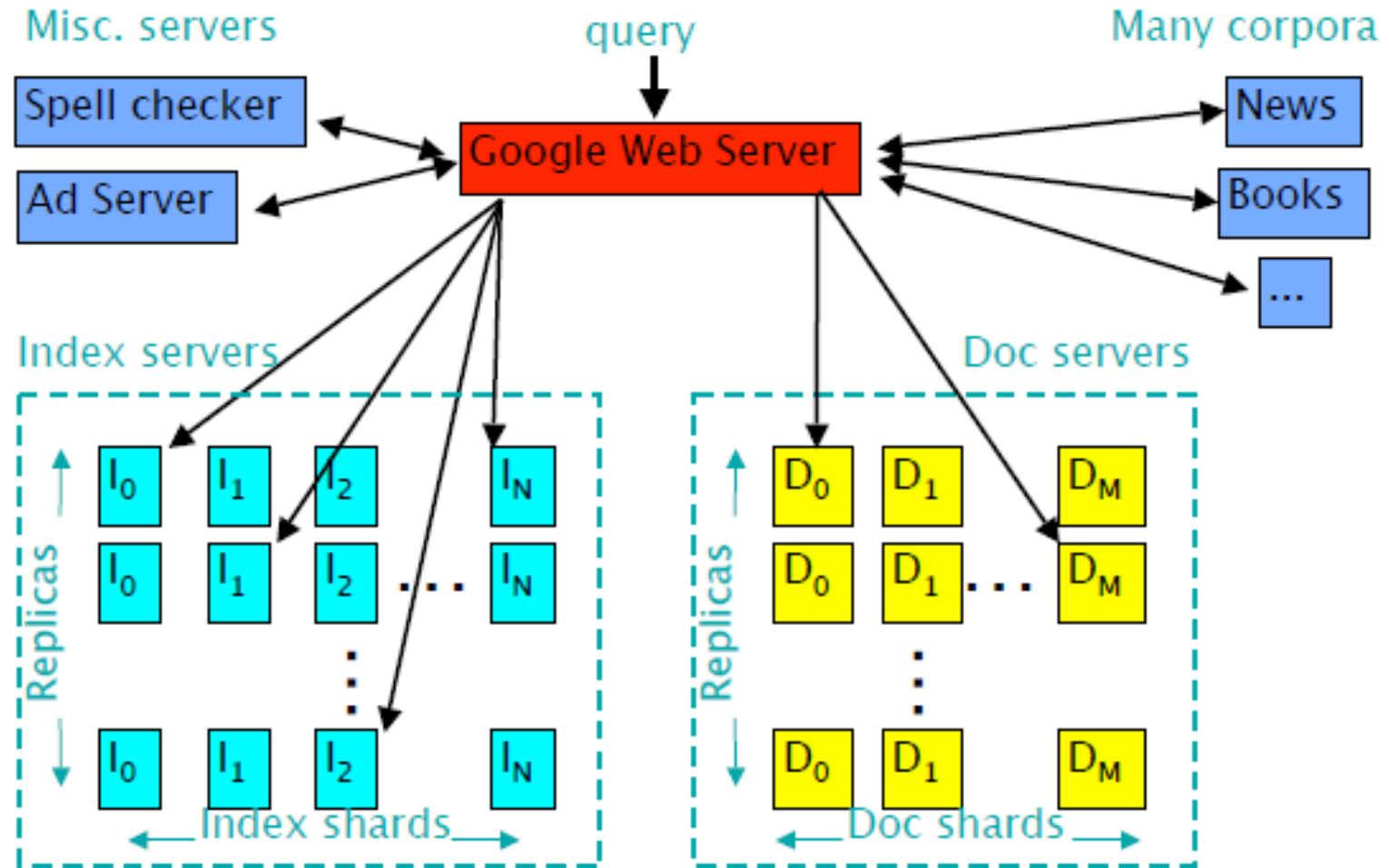
Google™ Cluster Architecture

Web Search for a Planet and much more....

Abhijeet Desai
desaiabhijeet89@gmail.com



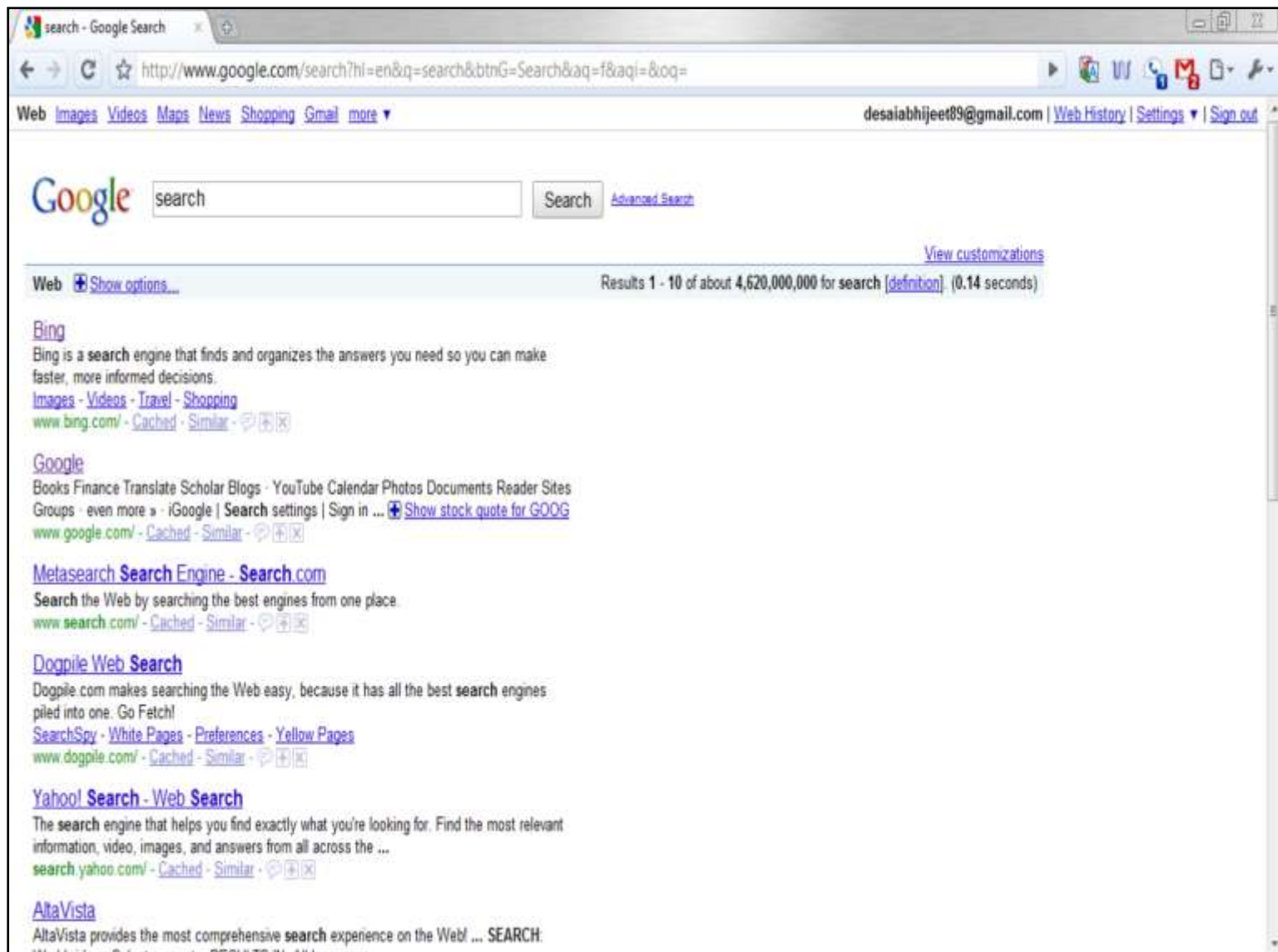
Google Query Serving Infrastructure



Elapsed time: 0.25s, machines involved: 1000s+

PageRank

- PageRank™ is the core technology to measure the importance of a page
- Google's theory
 - If page A links to page B
 - # Page B is important
 - # The link text is irrelevant
 - If many important links point to page A
 - # Links from page A are also important



Key Design Principles

- *Software reliability*
- *Use replication for better request throughput and availability*
- *Price/performance beats peak performance*
- *Using commodity PCs reduces the cost of computation*

The Power Problem

- High density of machines (racks)
 - High power consumption 400-700 W/ft²
 - # Typical data center provides 70-150 W/ft²
 - # Energy costs
 - Heating
 - # Cooling system costs
- Reducing power
 - Reduce performance (c/p may not reduce!)
 - Faster hardware depreciation (cost up!)

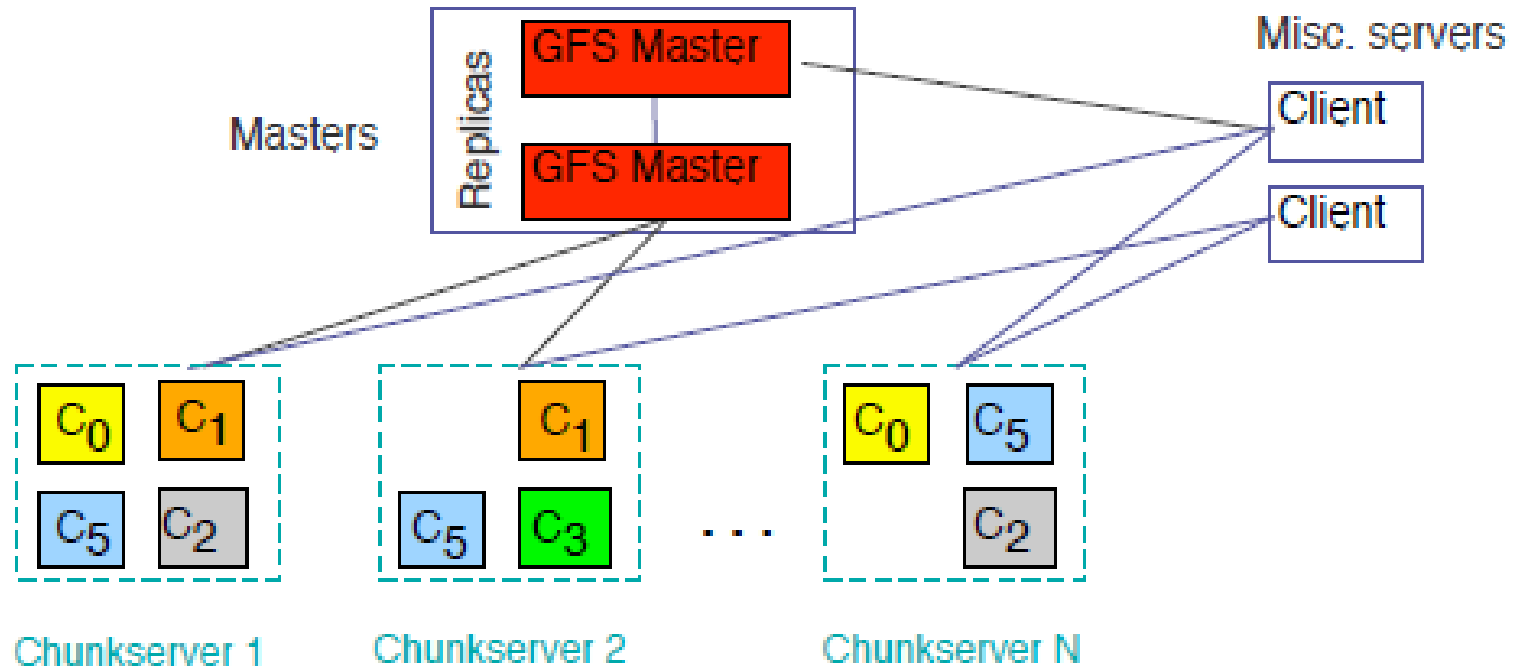
Parallelism

- Lookup of matching docs in a large index
--> many lookups in a set of smaller indexes followed by a merge step
- A query stream
--> multiple streams
(each handled by a cluster)
- Adding machines to a pool increases serving capacity

Hardware Level Consideration

- Instruction level parallelism does not help
- Multiple simple, in-order, short-pipeline core
- Thread level parallelism
- Memory system with moderate sized L2 cache is enough
- Large shared-memory machines are not required to boost the performance

GFS (Google File System) Design

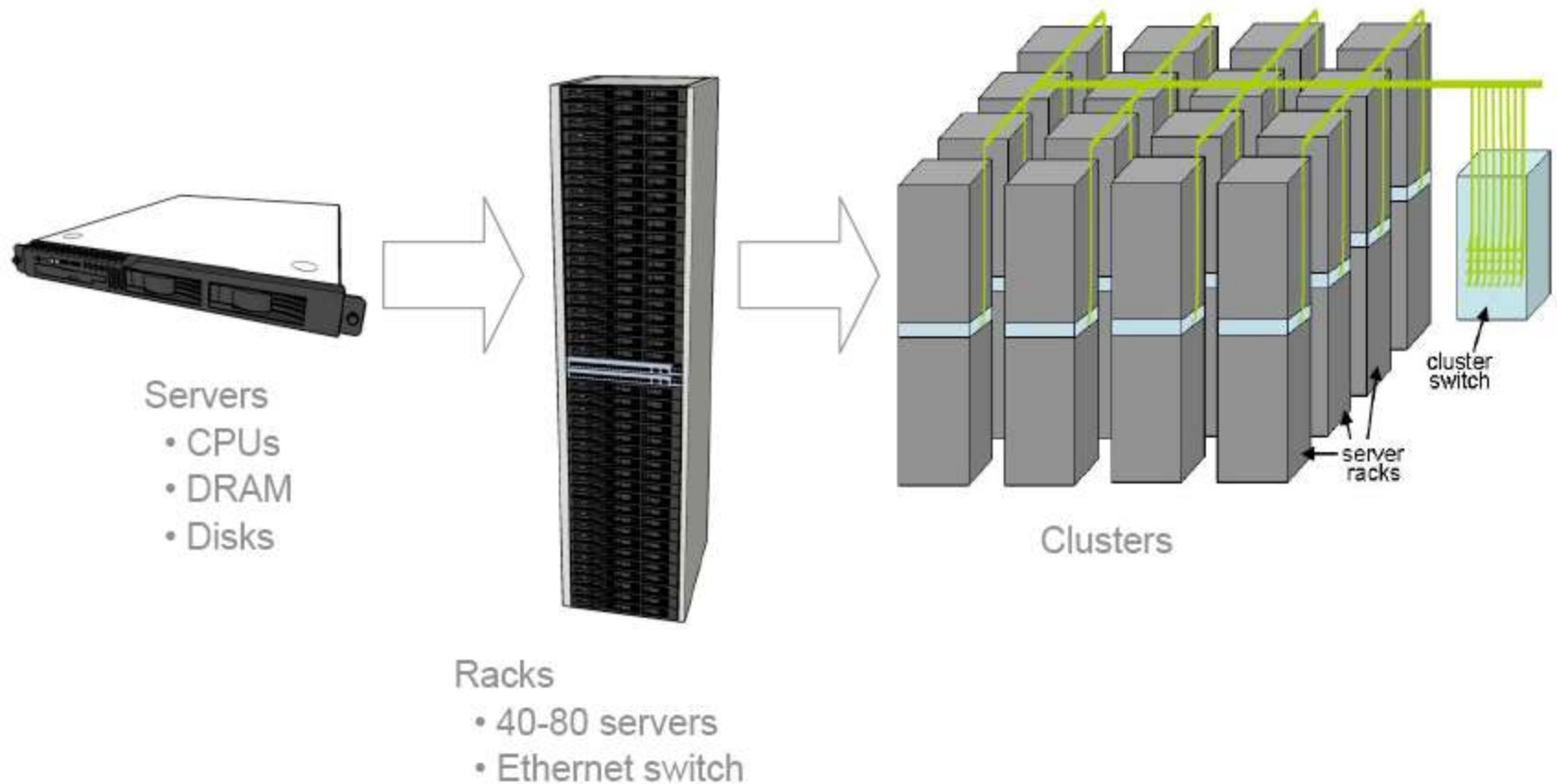


- Master manages metadata
- Data transfers happen directly between clients/chunk servers
- Files broken into chunks (typically 64 MB)

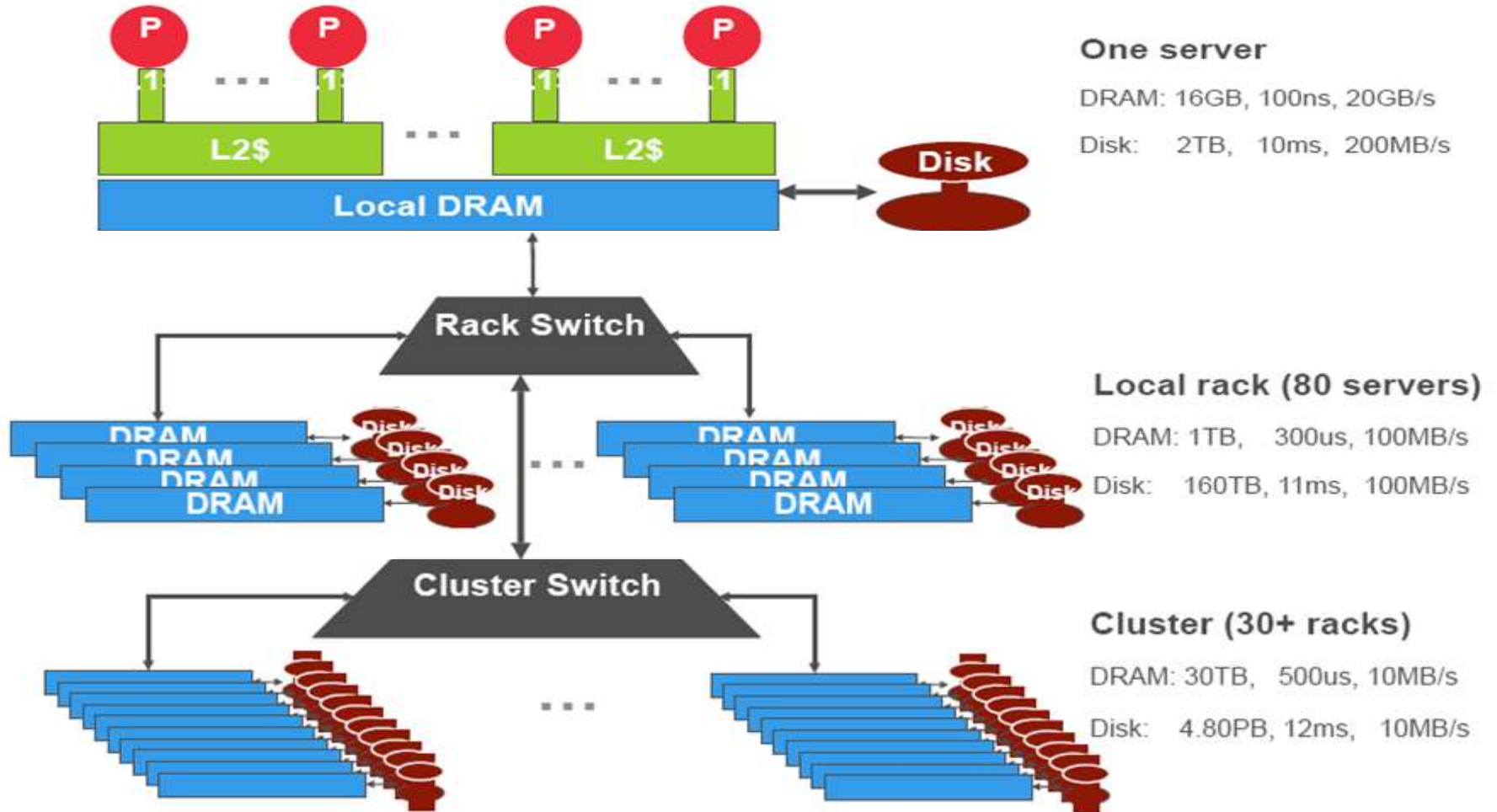
GFS Usage @ Google

- 200+ clusters
- Many clusters of 1000s of machines
- Pools of 1000s of clients
- 4+ PB Filesystems
- 40 GB/s read/write load
 - (in the presence of frequent HW failures)

The Machinery



Architectural view of the storage hierarchy

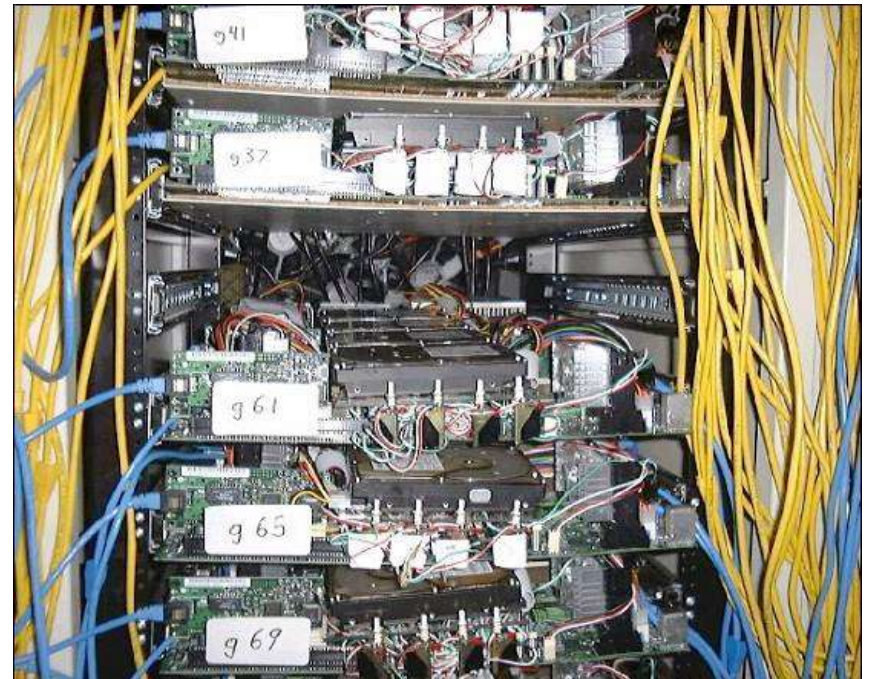


Clusters through the years

“Google” Circa 1997 (google.stanford.edu)



Google (circa 1999)



Clusters through the years

Google Data Center (Circa 2000)



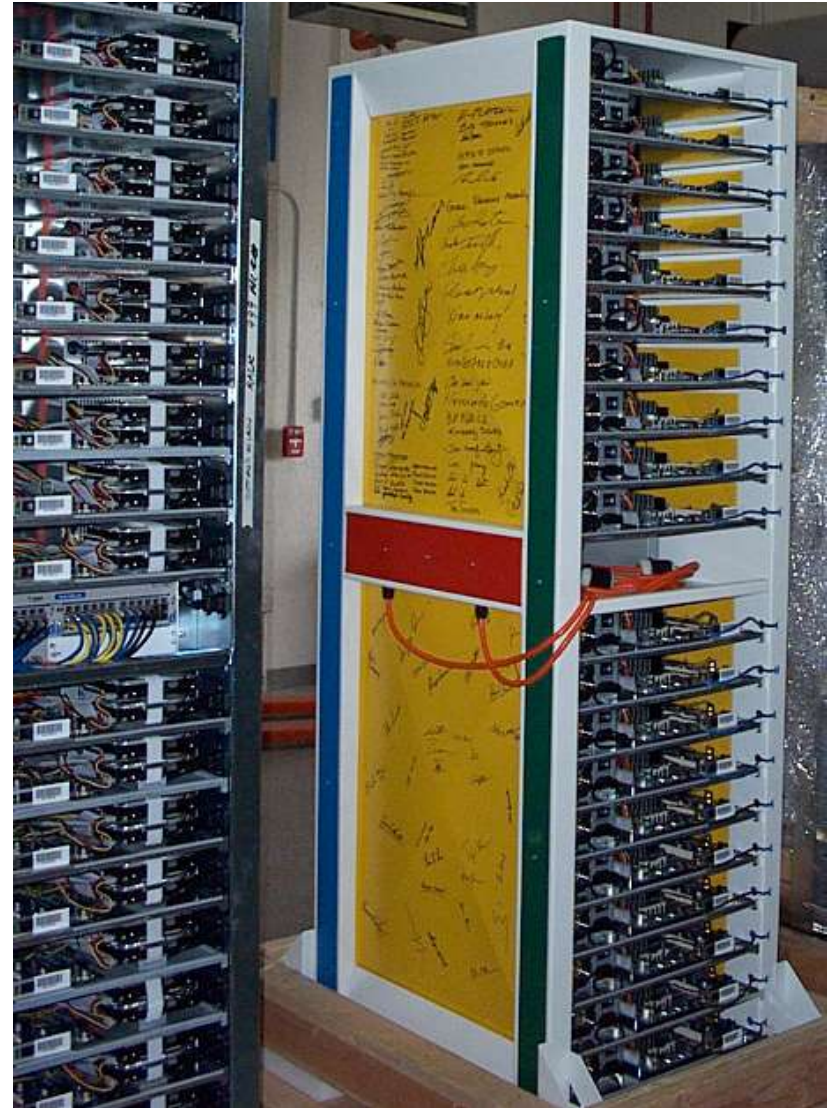
Google (new data center 2001)



3 days later

Current Design

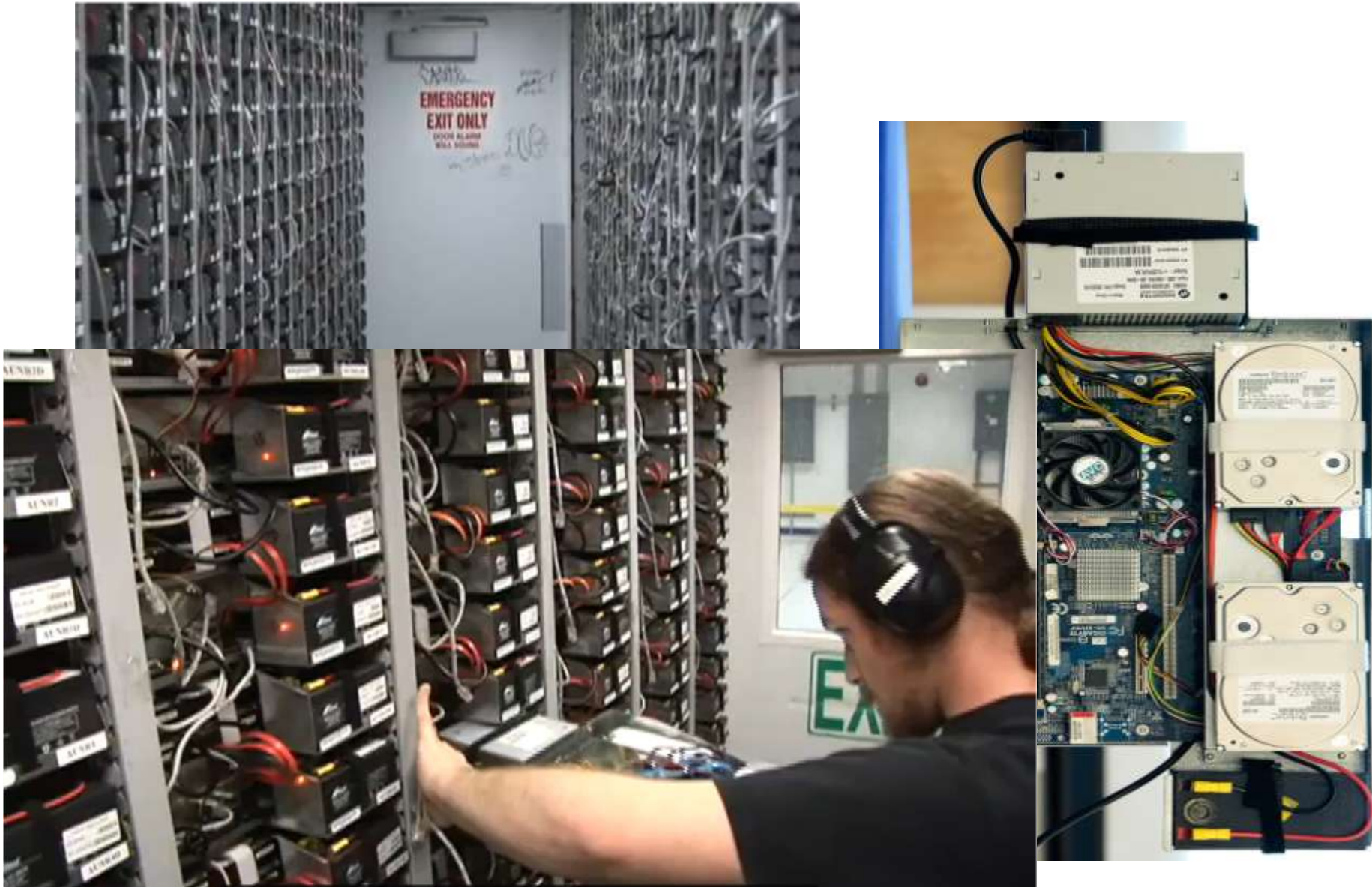
- In-house rack design
- PC-class motherboards
- Low-end storage and networking hardware
- Linux
- + in-house software



Container Datacenter



Container Datacenter



Multicore Computing



Comparison Between Custom built & High-end Servers

	Typical x86 - based server	Custom built x86 - based server	
PROCESSORS	8 2-GHz Xeon CPUs	176 2-GHz Xeon CPUs	22x
RAM	64 Gbytes of RAM	176 Gbytes of RAM	3x
DISK SPACE	8 Tbytes of disk space	7 Tbytes of disk space	-1 TB
PRICE	\$758,000	\$278,000	\$480,000

Implications of the Computing Environment

- *Stuff Breaks*
 - If you have one server, it may stay up three years (1,000 days)
 - If you have 10,000 servers, expect to lose ten a day
- *“Ultra-reliable” hardware doesn’t really help*
 - At large scale, super-fancy reliable hardware still fails, albeit less often
 - software still needs to be fault-tolerant
 - commodity machines without fancy hardware give better performance/\$
- **Reliability has to come from the software**
- **Making it easier to write distributed programs**

Infrastructure for Search Systems

- Several key pieces of infrastructure:
 - GFS
 - MapReduce
 - BigTable

MapReduce

- A simple programming model that applies to many large-scale computing problems
- Hide messy details in MapReduce runtime library:
 - automatic parallelization
 - load balancing
 - network and disk transfer optimizations
 - handling of machine failures
 - robustness
 - **improvements to core library benefit all users of library!**

Typical problem solved by MapReduce

- Read a lot of data
- **Map**: extract something you care about from each record
- Shuffle and Sort
- **Reduce**: aggregate, summarize, filter, or transform
- Write the results
- **Outline stays the same, map and reduce change to fit the problem**

Conclusions

- For a large scale web service system like Google
 - Design the algorithm which can be easily parallelized
 - Design the architecture using replication to achieve distributed computing/storage and fault tolerance
 - Be aware of the power problem which significantly restricts the use of parallelism

References

1. Luiz André Barroso , Jeffrey Dean , Urs Hölzle, Web Search for a Planet: The Google Cluster Architecture, IEEE Micro, v.23 n.2, p.22-28, March 2003 [doi>[10.1109/MM.2003.1196112](https://doi.org/10.1109/MM.2003.1196112)]
2. S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," Proc. Seventh World Wide Web Conf. (WWW7), International World Wide Web Conference Committee (IW3C2), 1998, pp. 107-117.
3. "TPC Benchmark C Full Disclosure Report for IBM eserver xSeries 440 using Microsoft SQL Server 2000 Enterprise Edition and Microsoft Windows .NET Datacenter Server 2003, TPC-C Version 5.0," <http://www.tpc.org/results/FDR/TPCC/ibm.x4408way.c5.fdr.02110801.pdf>.
4. D. Marr et al., "Hyper-Threading Technology Architecture and Microarchitecture: A Hypertext History," Intel Technology J., vol. 6, issue 1, Feb. 2002.
5. L. Hammond, B. Nayfeh, and K. Olukotun, "A Single-Chip Multiprocessor," Computer, vol. 30, no. 9, Sept. 1997, pp. 79-85.
6. L.A. Barroso et al., "Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing," Proc. 27th ACM Int'l Symp. Computer Architecture, ACM Press, 2000, pp. 282-293.
7. L.A. Barroso, K. Gharachorloo, and E. Bugnion, "Memory System Characterization of Commercial Workloads," Proc. 25th ACM Int'l Symp. Computer Architecture, ACM Press, 1998, pp. 3-14.